Question5

Solution :

Step 1 :

Compute the sum from T[1,1] to the T[n,1] and save the sum in a 2-dim array S as S[1,1].

Then delete T[1,1] and add T[n+1,1] and save the sum in a 2-dim array as S[1,2]. Process this column in T until delete T[3n-1,1] and plus T[4n,1] and save as S[3n,1].

At last, process each column like this column so we can get a 2-dim array S which dimension is [3n,3n].

Step 2:

Compute the sum from S[1,1] to the S[1,n] ,store the sum as maximum, this is the number of apples from an n*n matrix which's corner are {[1,1],[1,n+1],[n+1,1],[n+1,n+1]}. Store the index of first tree([1,1]) as matrix index.

Then delete S[1,1] and add S[1,n+1], and if the sum larger than the maximum, replace it and the index. Process this line in S like this.

At last, process each line like this line so we can get the largest total number of apples and the matrix index, if the index is S[i,j], the corner of square which contains the largest total number of apples are S[i,j], S[i+n,j], S[i,j+n], S[i+n,j+n].

Proof :

In the first step, the time complexity:

Compute one column: $n + 2(2n + 1) = O(n)$

Compute all column: $O(n^2)$

In the second step, the time complexity:

Compute one line: $n + 2(2n + 1) = O(n)$

Compute all line: $O(n^2)$

Total time complexity:

$O(n^2) + O(n^2) = O(n^2)$